Resim1: Album Sayfası Örneği

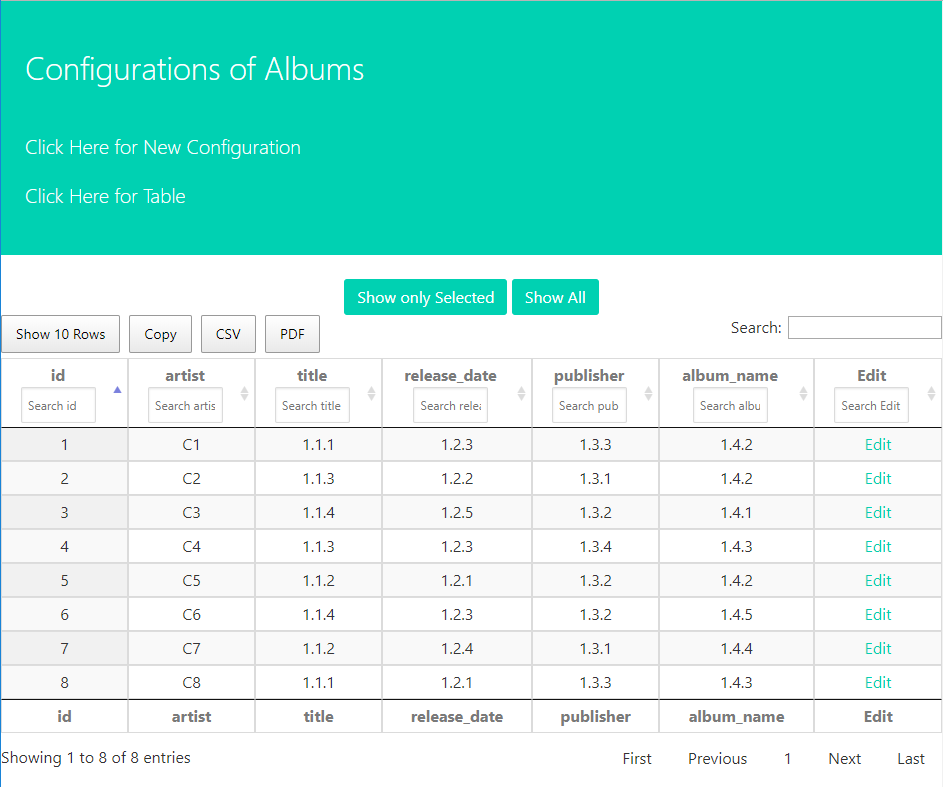
Versiyon Takip Sistemi

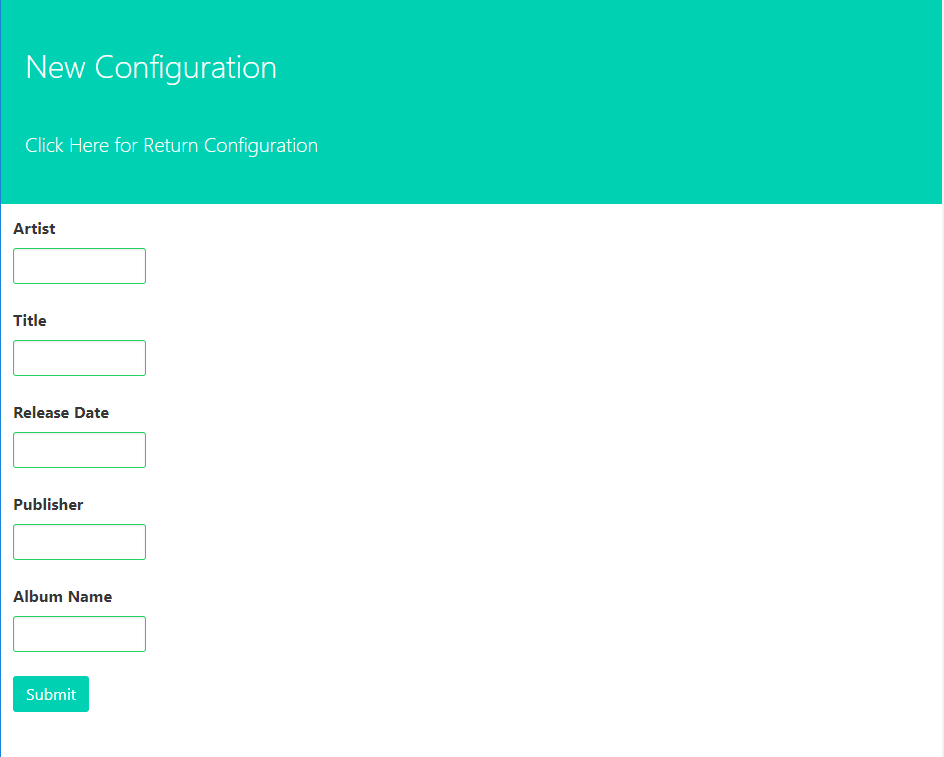
Bu uygulamanın amacı uçaklarda bulunan farklı yazılımların versiyonlarının kontrollerinin rahat bir biçimde yapılabilmesi ve en çok hangi konfıgürasyon tipine benziyorsa onun görülebilinmesi. Uygulamanın genel olarak özellikleri aşağıdadır. Daha detaylı açıklama ilerideki bölümlerde mevcuttur.

* Konfigürasyon ekleme-kaldırma-düzenleme
* Yeni uçak ekleme-kaldırma-düzenleme
* Data tablosunda sütun bazlı arama yapabilme
* Data tablosunda sütun bazlı sıralama yapabilme
* Data tablosunda seçilen satırları ayrı olarak listeleyebilme.
* Data tablosunu CSV veya PDF formatında çıktısını alabilmek bunun yanında clipboard’a kopyalayabilmek.

Uygulama Python Flask, PosgreSQL, HTML, CSS, Jquery, JS kullanarak geliştirilmiştir. Datatables.net sitesinden de yardım alınmış ve özellikleri kullanılmıştır.

Uygulama Örnek Sayfaları:

Resim 2: Konfigürasyon Sayfası Örneği

  
Resim 3: Satır Düzenleme-Ekleme Sayfası Örneği

Uygulama ilk kez çalıştırılmadan önce yüklenmesi gerekenler:

1. Python en düşük 3.6 sürümü
2. Postgresql
3. Virtual Environment oluşturulması gerekiyor windowsta doğru çalışabilmesi için

Uygulamayı çalıştırırken yapılması gerekenler:

Uygulama için gerekli olan tüm kütüphaneler dosyanın içindeki dbenv klasörünün içinde bulunuyor. Çalıştırırken uygulanması gereken adımlar aşağıdaki şekildedir:

1. Virtual environment(dbenv)'i çalıştırmak.(dbenv\Scripts\activate). Çalışmaya başladığını komut satırının en solunda '(dbenv)' görerek anlayabilirsiniz.

2. Sonrasında 'py <Dosya İsmi>' şeklindeki komut ile sırasıyla aşağıdaki dosyalar çalıştırılır.

1. db\_setup.py
2. db\_creator.py
3. forms.py
4. models.py
5. app.py
6. main.py

3. main.py dosyasının çalışmasıyla beraber server aktif hale gelir ve aynı zamanda database e bağlanmış olur.

4-Bu işlemler bir bilgisayarda bir kez yapılmışsa direkt main.py klasörünün çalıştırılması yeterli olacaktır.

Program çalışmaya başladı. Programın genel olarak özellikleri şunlardır:

1. Datatable ve Configuration tabledan oluşur.
2. Datatable'a her eleman eklemesi veya düzeltilmesinde en çok benzediği konfigurasyonu gösteren sütündaki eleman da güncellenir. Tek tek tüm konfigürasyonlarla karşılaştırarak en çok sayıda elemanının benzediği konfigürasyonu yanına yazar.
3. Configuration Table'a her eleman eklenmesi veya düzeltilmesinde Datatable'daki tüm elemanların konfigürasyonunu gösteren elemanlarını tek tek karşılaştırarak tekrar düzenler.
4. Tablo üzerinde sütun bazlı sıralama yapılabilir. Eğer birden fazla sütuna göre sıralanmak istenirse 'Shift' tuşuna bastıktan sonra sütunlara basmak gerekir.
5. Sütun bazlı arama yapılabilir.Aynı anda bir kaç sütunda aratma yapabilirsiniz.
6. Dosya PDF-CSV şeklinde dışarı aktarılabilir. Aynı zamanda komple kopyalanabilir. CSV şeklinde kaydetmede ise ayıran kısmında diğer seçeneğini seçerek <'> işaretini ve metin ayracı kısmında ise <"'"> işaretini eklemek gerekiyor. Bunun sonrasında elde sütun sütun ayrılmış tablo ortaya çıkıyor.
7. Show 10 Rows seçeneğinde kaç tane elemanın gösterileceği seçilebilir.
8. Sıralama veya arama dışında özellikle seçmek istenen satırlar olduğunda satırların üstüne tek tek tıkladıktan sonra 'Show only Selected' butonuna tıklayarak seçilen satırlar gösterilebilir. 'Show All' butonuna tıklayarak da tekrar tüm sütunlar görülebilir.

Genel hatlarıyla dosyalar:

1. db\_setup.py: engine = create\_engine satırının devamında postgresql://<kullanıcıadı>:<şifre>@localhost/<proje\_ismi> şeklinde istenilen database e bağlanabilir. Bu yapılan değişikliğin db\_creator.py ve app.py dosyalarının ilgili satırlarında da yapmak gerekiyor.
2. db\_creator.py: Bu dosya databasede istenilen tablo ve değişkenlerin oluşturulmasını sağlar. Yeni tablo eklenmek istendiğinde eklenmiş 2 tablo gibi yazarak eklenebilir. Yeni sütun eklenmek istendiğinde ise tekrar bir satır daha eklenebilir ancak models.py ve forms.py'daki değişiklikler unutulmamalı. Halihazırda var olan bir tablo yeni sütun db\_creator.py klasöründen eklenmeye çalışılırsa bu gerçekleşmez. Tablonun ismi değişilip yeni tablo olarak oluşturulması lazım. Veya psql(postgresql paneli) kullanılarak oluşturulabilir. Ancak her satırdaki veriye o sütun için veri eklemek gerekir. En iyi seçenek yeni tablo oluşturulup bu tabloda bulunan verilerin oraya aktarılmasıdır.
3. forms.py: Bu dosya yeni dosya ekleme ve editleme sayfalarındaki formdaki verileri bir değişken olarak belirlemek için var. Yeni tablo eklendiğinde eğer verilerin editlenmesi veya eklenebilmesi isteniyorsa buraya da o tablonun bilgilerinin eklenmesi gerekiyor.
4. models.py: Bu dosya database'e kaydettiğimiz tabloları model olarak tutar ve kullanımı kolaylaştırır. Yeni bir tablo eklendiğinde bu dosyada da ekleme yapılması gerekiyor.
5. app.py: Bu dosya database ile uygulama arasındaki bağlantıyı sağlıyor. secret\_key koymak gerekiyor ancak sorulan veya tekrar o şifrenin girildiği hiçbir yer yok.
6. main.py: Programda kullanıcı gözünde gerçekleşen tüm olayların arka planını bu dosya sağlıyor. Genel olarak siteden verileri veya istekleri alıp, onu işleyip yeni bir siteye yönlendiriyor. Fonksiyon olarak bakmak gerekirse:
7. index(): Datatable'ı results.html dosyasına göndererek onun çalışmasını sağlar.
8. configurations(): Configuration tablosunu configurations.html dosyasına göndererek çalışmasını sağlar
9. new\_album(): Datatable'a yeni bir dosya eklenmek istendiğinde bu fonksiyon çalışarak '/new\_album' uzantılı bağlantıya gider ve burada girilen formu alır. Bu formu ve oluşturduğu album değişkenini save\_changes() fonksiyonuna gönderir.
10. save\_changes(): Dosyanın yeni dosya mı yoksa editlenecek dosya mı olduğuna bakarak database'e ekler veya sadece commitler. Eşitleme kısmında iste album ve form değişkenlerini üzerinde daha kolay gezebilmek için dictionary'e çevirir. Sonrasında ise key değerlerini gezerek formdan gelen kullanıcı verisiyle tablo verisini birbirlerine eşitler. configuration\_type() fonksiyonundan gelen 'name' değişkenini ise eşitleyerek onu belirlemiş olur. setattr(album, key, value) fonksiyonu album.key=value şeklinde değiştirir.
11. configuration\_type(): Aldığı form verisininin elemanlarını tek tek tüm configurations tablosundaki tüm elemanlarla karşılaştırır ve en çok benzeyen elemanı olan konfigürasyonu return ederek tablodaki bu elemanı gösteren değişkene eşitler.
12. new\_configuration(): new albumde yapılanın aynısı yapılır ancak bu kez save\_changes\_configuration() fonksiyonuna gönderilir.
13. save\_changes\_configuration(): Bunun save\_changes() fonksiyonundan tek farkı configuration\_type değişkenine ihtiyaç duymaması. Ayrıca en sonda configurasyona yeni bir şey eklendiği veya değiştirildiği için datatable'ın tamamen güncellenmesi için reload\_table() fonksiyonunu çalıştırır.
14. reload\_table(): datatable'daki tüm elemanları gezerek onların konfigürasyon tipini tutan elemanının güncellenip güncellenmediğine bakmak için configuration\_type() fonksiyonuna gönderir. Değişiklikleri database e ekler.
15. edit(): new\_album ile aynı sayfa açılır ancak bunun farkı tablodaki verilerin kutucuklara dolu olarak gelmesidir. Sonrasında da new\_album ile aynı adımlar işlenir.
16. edit\_configuration(): new\_configuration ile aynı sayfa açılır ancak bunun farkı tablodaki verilerin kutucuklara dolu olarak gelmesidir. Sonrasında da new\_configuration ile aynı adımlar işlenir.
17. object\_as\_dict(): Üretilmiş olan album-configuration objeleri dictionary'e çevirmek için bulunur.

HTML DOSYALARI

1. result.html - configurations.html: Datatable'ın kullanıcıya gözüktüğü sayfadır. Genel olarak datatable.net sitesindeki özellikler kullanılmıştır ancak eklemeler vardır. Kod 3 kısımdan oluşmaktadır ve bu kısımlar sayfa ismi ve yönlendirme linklerinin farkı hariç configurations.html dosyasındakiyle aynıdır. Sayfanın tasarımı genel olarak bulma framework ile yapıldı. bulma.css dosyası dahil edildikten sonra kullanılmak istenen kısımların class kısmını özel kelimeler kullandıktan sonra kalan işi bulma kendisi yapıyor. CSS dosyası yazmaya gerek kalmıyor. (Örn.<button class="btn btn-primary button is-primary" id="btn-show-selected"> Show only Selected</button> 'button is-primary' kelimesi şekli ve rengini ayarlamasına yetiyor.)

A.1) Dahil edilen CSS(Cascading Style Sheets=), JS(Java Script) dosyalarınının proje dosyası içindeki konumları.

A.2) Tabloda gerçekleşen tıklanmalar sonucu yapılacak şeyleri düzenleyen JS kodları. Bu kısımdaki fonksiyonların ve değişkenlerin açıklamaları dosya içinde yorum satırı olarak eklenmiştir. Genel anlamda tablo üzerinde gerçekleşen arama, seçip listeleme, pdf-csv gibi çıktılar veren butonlar, üzerinden geçilen sütunun renk değişmesi, gösterilecek satır sayısı vs.

A.3) Kullanıcının gördüğü yazı ve tabloların azıldığı HTML(HyperText Markup Language) kısmı. Burada main.py dosyasının içindeki index fonksiyonunda results.html çağırılırken kullanılan table değişkeni ile beraber geliyor. Tablo yazdirilirken de tek tek bu tabloyu yazmak yerine döngü ile beraber tüm tablo yazdırılıyor.

B) edit\_album.html - edit\_configuration.html - new\_album.html - new\_configuration.html: \_formhelpers.html dosyasından verileri alarak formları ekrana yazdırır. editlemek için olan sayfalarda ise kutucukların içi halihazırda bulunan verilerle dolu olarak gelir. Sitenin tasarım kısmı ise Bulma CSS Framework ile yapılmıştır.